

This listing of claims will replace all prior versions, and listings, of claims in the application.

**Listing of Claims:**

Claim 1 (Currently Amended): A method for synchronizing a plurality of instances for a data platform, ~~including without limitation a replica or a data source with or without an adapter, and said plurality of instances comprising a syne community,~~ said method comprising:

uniquely enumerating changes in sequence (~~e.g., increasing numbers~~) on an instance by instance basis (~~a change number~~) using a change number;

maintaining a vector for each instance, said vector comprising a most recent change number for said instance, ~~as well as~~ and a most recent known change number for each other instances from among said plurality of instances that are known to said instance, each being a known instance (~~each a known instance~~), wherein said vector (~~comprising change numbers for each known instance~~) inherently represents all changes that have been made to said instance.

Claim 2 (Original): The method of claim 1 wherein said change number comprises a unique identification number for said instance and a relative incremental count of changes made to said instance.

Claim 3 (Original): The method of claim 1 wherein a first instance, to partially synchronize with a second instance, requests changes from said second instance by sending to said second instance its vector, and wherein said second instance, based on the vector it receives from said first instance, sends to said first instance only those changes that said first vector has not yet received.

Claim 4 (Currently Amended): The method of claim 3 wherein said second instance, based on the vector it receives from said first instance, further determines that said first instance has changes that ~~[[it,]]~~ said second instance~~[[,]]~~ has not yet received, and sends its own vector to said first instance to request these changes, and wherein said first instance, based on the vector it receives from said second instance, sends to said second instance only those changes that said second vector has not yet received.

Claim 5 (Currently Amended): The method of claim 1 wherein a first instance, when changing a first Item to relate via a Relationship (~~via a Relationship~~) to a second Item that was not previously being synchronized (~~e.g., an out of scope Item~~), to send all change information pertaining to said second Item to a second instance when synchronizing with said second instance so that said second Item in said second instance is synchronized with said second Item in said first instance.

Claim 6 (Currently Amended): The method of claim 1 wherein, when by default operation a parent Item is ~~transmitted~~ transmitted before a child Item to said parent Item, and wherein a plurality of changes are typically transmitted in sequential order of a plurality of change numbers corresponding to said changes during a synchronization, and wherein between synchronizations a child Item is changed and then a parent Item is changed in a first instance, any change units pertaining to said child Item are sent from said first instance to a second instance during a synchronization only after all change units pertaining to the parent Item are sent.

Claim 7 (Original): The method of claim 1 wherein, for an Item deleted by a first instance, a tombstone comprising the identification of the Item deleted is created, and said tombstone is sent as part of a synchronization to notify a second instance to identify the Item to be deleted in said second instance.

Claim 8 (Currently Amended): The method of claim 7 wherein, for a first instance having a parent Item and a first child Item to said parent Item, when said child Item is deleted and then said parent Item is deleted, said first instance sending only the change to said parent Item (~~that is, the tombstone for said parent Item~~) to a second instance during a synchronization where (a) the deletion of a parent Item automatically results in the deletion of all children Items for said parent or (b) the second instance, receiving the tombstone for the parent Item, proceeds to delete the parent Item and automatically deletes the child Item.

Claim 9 (Currently Amended): The method of claim 1, wherein a first Relationship and a second Relationship of a first instance swap names using a temporary name element such that, in order, (a) the name of the first Relationship is transferred to said temporary name element, (b) the name of the second Relationship is transferred to said first Relationship, and (c) said name stored in the temporary name element is copied to said second Relationship, and wherein said first instance synchronizes with a second instance and sends a duo of change units representing, in order, (i) the new name for said first Relationship and (ii) the new name for said second Relationship, and wherein effecting the first change of said duo of changes (~~a first change~~) results in an attempted change having an error in the second instance because a result of said first change is for the first Relationship and the second Relationship having the same name (~~an attempted change~~), a method by which said second instance proceed to copy said name of the first Relationship to a local temporary name element and:

if, during the synchronization, a subsequent change is received for copying the name of said second Relationship to said first relationship ~~occurs~~, then performing said change as well as also copying said name in said local temporary name element to said first Relationship; and

if, during the synchronization, a subsequent change is not received for copying the name of said second Relationship to said first relationship ~~occurs~~, then raising a conflict regarding for the attempted change.

Claim 10 (Currently Amended): The method of claim 1 wherein, for synchronization between a first instance on a storage platform that allows a dangling relative reference and a second instance on a storage platform that does not allow a dangling relative reference that include at least one change to a relative reference and at least one other change (~~in that order~~), sending said change to said relative references after said one other changes.

Claim 11 (Currently Amended): A system implemented at least in part by a computing device for synchronizing a plurality of instances for a data platform, ~~including without limitation a replica or a data source with or without an adapter, and said plurality of instances comprising a sync community~~, said system comprising:

a subsystem for uniquely enumerating changes in sequence (~~e.g., increasing numbers~~) on an instance by instance basis (~~a change number~~) using a change number;

a subsystem for maintaining a vector for each instance, said vector comprising a most recent change number for said instance, ~~as well as~~ and a most recent known change number for each other instances from among said plurality of instances that are known to said instance, each being a known instance (~~each a known instance~~), wherein said vector (~~comprising change numbers for each known instance~~) inherently represents all changes that have been made to said instance.

Claim 12 (Original): The system of claim 11 wherein said change number comprises a unique identification number for said instance and a relative incremental count of changes made to said instance.

Claim 13 (Original): The system of claim 11 further comprising a subsystem wherein a first instance, to partially synchronize with a second instance, requests changes from said second instance by sending to said second instance its vector, and wherein said second instance, based on the vector it receives from said first instance, sends to said first instance only those changes that said first vector has not yet received.

Claim 14 (Currently Amended): The system of claim 13 further comprising a subsystem wherein said second instance, based on the vector it receives from said first instance, further determines that said first instance has changes that [[it,]] said second instance~~[[,]]~~ has not yet received, and sends its own vector to said first instance to request these changes, and wherein said first instance, based on the vector it receives from said second instance, sends to said second instance only those changes that said second vector has not yet received.

Claim 15 (Currently Amended): The system of claim 11 further comprising a subsystem wherein a first instance, when changing a first Item to relate via a Relationship (~~via a Relationship~~) to a second Item that was not previously being synchronized (~~e.g., an out-of-scope Item~~), to send all change information pertaining to said second Item to a second

instance when synchronizing with said second instance so that said second Item in said second instance is synchronized with said second Item in said first instance.

Claim 16 (Currently Amended): The system of claim 11 further comprising a subsystem wherein, when by default operation a parent Item is ~~transmitted~~ transmitted before a child Item to said parent Item, and wherein a plurality of changes are typically transmitted in sequential order of a plurality of change numbers corresponding to said changes during a synchronization, and wherein between synchronizations a child Item is changed and then a parent Item is changed in a first instance, any change units pertaining to said child Item are sent from said first instance to a second instance during a synchronization only after all change units pertaining to the parent Item are sent.

Claim 17 (Original): The system of claim 11 further comprising a subsystem wherein, for an Item deleted by a first instance, a tombstone comprising the identification of the Item deleted is created, and said tombstone is sent as part of a synchronization to notify a second instance to identify the Item to be deleted in said second instance.

Claim 18 (Currently Amended): The system of claim 17 further comprising a subsystem wherein, for a first instance having a parent Item and a first child Item to said parent Item, when said child Item is deleted and then said parent Item is deleted, said first instance sending only the change to said parent Item (~~that is, the tombstone for said parent Item~~) to a second instance during a synchronization where (a) the deletion of a parent Item automatically results in the deletion of all children Items for said parent or (b) the second instance, receiving the tombstone for the parent Item, proceeds to delete the parent Item and automatically deletes the child Item.

Claim 19 (Currently Amended): The system of claim 11, further comprising a subsystem wherein a first Relationship and a second Relationship of a first instance swap names using a temporary name element such that, in order, (a) the name of the first Relationship is transferred to said temporary name element, (b) the name of the second Relationship is transferred to said first Relationship, and (c) said name stored in the temporary name element

is copied to said second Relationship, and wherein said first instance synchronizes with a second instance and sends a duo of change units representing, in order, (i) the new name for said first Relationship and (ii) the new name for said second Relationship, and wherein effecting the first change of said duo of changes (~~a first change~~) results in an attempted change having an error in the second instance because a result of said first change is for the first Relationship and the second Relationship having the same name (~~an attempted change~~), a method by which said second instance proceed to copy said name of the first Relationship to a local temporary name element and:

if, during the synchronization, a subsequent change is received for copying the name of said second Relationship to said first relationship ~~occurs~~, then performing said change as well as also copying said name in said local temporary name element to said first Relationship; and

if, during the synchronization, a subsequent change is not received for copying the name of said second Relationship to said first relationship ~~occurs~~, then raising a conflict regarding for the attempted change.

Claim 20 (Currently Amended): The system of claim 11 further comprising a subsystem wherein, for synchronization between a first instance on a storage platform that allows a dangling relative reference and a second instance on a storage platform that does not allow a dangling relative reference that include at least one change to a relative reference and at least one other change (~~in that order~~), sending said change to said relative references after said one other changes.

Claim 21 (Currently Amended): A computer-readable storage medium comprising computer-readable instructions for synchronizing a plurality of instances for a data platform, ~~including without limitation a replica or a data source with or without an adapter, and said plurality of instances comprising a sync community~~, said computer-readable instructions comprising instructions for:

uniquely enumerating changes in sequence (~~e.g., increasing numbers~~) on an instance by instance basis (~~a change number~~) using a change number;

maintaining a vector for each instance, said vector comprising a most recent change number for said instance, ~~as well as~~ and a most recent known change number for each other instances from among said plurality of instances that are known to said instance, each being a known instance (~~each a known instance~~), wherein said vector (~~comprising change numbers for each known instance~~) inherently represents all changes that have been made to said instance.

Claim 22 (Currently Amended): The computer-readable ~~instructions~~ storage medium of claim 21 further comprising instructions for said change number to comprises a unique identification number for said instance and a relative incremental count of changes made to said instance.

Claim 23 (Currently Amended): The computer-readable ~~instructions~~ storage medium of claim 21 further comprising instructions for partially synchronizing a first instance with a second instance with said first instance requesting changes from said second instance by sending to said second instance its vector, and wherein said second instance, based on the vector it receives from said first instance, sends to said first instance only those changes that said first vector has not yet received.

Claim 24 (Currently Amended): The computer-readable ~~instructions~~ storage medium of claim 23, wherein said second instance, based on the vector it receives from said first instance, further comprises instructions for said second instance to determine that said first instance has changes that ~~[[it,]]~~ said second instance~~[[,]]~~ has not yet received, and sends its own vector to said first instance to request these changes, and wherein said first instance, based on the vector it receives from said second instance, sends to said second instance only those changes that said second vector has not yet received.

Claim 25 (Currently Amended): The computer-readable ~~instructions~~ storage medium of claim 21 further comprising instructions for a first instance, when changing a first Item to relate via a Relationship (~~via a Relationship~~) to a second Item that was not previously being synchronized (~~e.g., an out-of-scope Item~~), to send all change information pertaining to said

second Item to a second instance when synchronizing with said second instance so that said second Item in said second instance is synchronized with said second Item in said first instance.

Claim 26 (Currently Amended): The computer-readable ~~instructions~~ storage medium of claim 21 further comprising instructions for wherein, when by default operation a parent Item is ~~transmitted~~ transmitted before a child Item to said parent Item, and wherein a plurality of changes are typically transmitted in sequential order of a plurality of change numbers corresponding to said changes during a synchronization, and wherein between synchronizations a child Item is changed and then a parent Item is changed in a first instance, any change units pertaining to said child Item are sent from said first instance to a second instance during a synchronization only after all change units pertaining to the parent Item are sent.

Claim 27 (Currently Amended): The computer-readable ~~instructions~~ storage medium of claim 21 further comprising instructions for wherein, for an Item deleted by a first instance, a tombstone comprising the identification of the Item deleted is created, and said tombstone is sent as part of a synchronization to notify a second instance to identify the Item to be deleted in said second instance.

Claim 28 (Currently Amended): The computer-readable ~~instructions~~ storage medium of claim 27 further comprising instructions for wherein, for a first instance having a parent Item and a first child Item to said parent Item, when said child Item is deleted and then said parent Item is deleted, said first instance sending only the change to said parent Item (~~that is, the tombstone for said parent Item~~) to a second instance during a synchronization where (a) the deletion of a parent Item automatically results in the deletion of all children Items for said parent or (b) the second instance, receiving the tombstone for the parent Item, proceeds to delete the parent Item and automatically deletes the child Item.

Claim 29 (Currently Amended): The computer-readable ~~instructions~~ storage medium of claim 21 further comprising instructions for, wherein a first Relationship and a second



Relationship of a first instance swap names using a temporary name element such that, in order, (a) the name of the first Relationship is transferred to said temporary name element, (b) the name of the second Relationship is transferred to said first Relationship, and (c) said name stored in the temporary name element is copied to said second Relationship, and wherein said first instance synchronizes with a second instance and sends a duo of change units representing, in order, (i) the new name for said first Relationship and (ii) the new name for said second Relationship, and wherein effecting the first change of said duo of changes (~~a first change~~) results in an attempted change having an error in the second instance because a result of said first change is for the first Relationship and the second Relationship having the same name (~~an attempted change~~), a method by which said second instance proceed to copy said name of the first Relationship to a local temporary name element and:

if, during the synchronization, a subsequent change is received for copying the name of said second Relationship to said first relationship ~~occurs~~, then performing said change as well as also copying said name in said local temporary name element to said first Relationship; and

if, during the synchronization, a subsequent change is not received for copying the name of said second Relationship to said first relationship ~~occurs~~, then raising a conflict regarding for the attempted change.

Claim 30 (Currently Amended): The computer-readable ~~instructions~~ storage medium of claim 21 further comprising instructions for wherein, for synchronization between a first instance on a storage platform that allows a dangling relative reference and a second instance on a storage platform that does not allow a dangling relative reference that include at least one change to a relative reference and at least one other change (~~in that order~~), sending said change to said relative references after said one other changes.